Robust model-based tracking for robot vision

Andrew Comport, E. Marchand, François Chaumette

IEEE/RSJ Int. Conf on Inteligent Robots and Systems, IROS'04, Sendai, Japan, September 2004.

Robust model-based tracking for robot vision

Andrew I. Comport, Éric Marchand, François Chaumette
IRISA - INRIA Rennes
Campus de Beaulieu, 35042 Rennes, France
E-Mail : Firstname.Lastname@irisa.fr

***Abstract*— This paper proposes a real-time, robust and efficient 3D model-based tracking algorithm for visual servoing. A virtual visual servoing approach is used for monocular 3D tracking. This method is similar to more classical non-linear pose computation techniques. A concise method for derivation of efficient distance-to-contour interaction matrices is described. An oriented edge detector is used in order to provide real-time tracking of points normal to the object contours. Robustness is obtained by integrating a M-estimator into the virtual visual control law via an iteratively re-weighted least squares implementation. The method presented in this paper has been validated on several 2D 1/2 visual servoing experiments considering various objects. Results show the method to be robust to occlusion, changes in illumination and miss-tracking.**

I. INTRODUCTION

This paper addresses the problem of robust real-time model-based tracking of 3D objects by employing virtual visual servoing techniques. This fundamental vision problem has applications in many domains ranging from visual servoing [14], [8], [12] to medical imaging and robotics or industrial applications. Most of the available tracking techniques can be divided into two main classes: feature-based and model-based. The former approach focuses on tracking 2D features such as geometrical primitives (points, segments, circles,...), object contours [15], regions of interest [10]...The latter explicitly uses a model of the tracked objects. This can be a CAD model [9], [17], [19], [7] or a 2D template of the object [16]. This second class of methods usually provides a more robust solution. The main advantage of the model-based methods is that the knowledge about the scene (the implicit 3D information) allows improvement of robustness and performance by being able to predict hidden movement of the object, detect partial occlusions and acts to reduce the effects of outlier data introduced in the tracking process. In this paper a model-based algorithm is proposed for the tracking of 3D objects whose CAD model is known.

This study focuses on the registration techniques that allow alignment of real and virtual worlds using images acquired in real-time by a moving camera. In the related computer vision literature geometric primitives considered for the estimation are often points [11], [5], [20], contours or points on the contours [19], [3], [7], segments, straight lines, conics, cylindrical objects, or a combination of these different features [21]. Another important issue is the registration problem. *Purely geometric* (eg, [6]), or *numerical and iterative* [5] approaches may be considered. *Linear approaches* use a least-squares method to estimate the pose. *Full-scale non-linear optimization techniques* (e.g., [19], [7]) consist of minimizing the error between the observation and the forward-projection of the model. In this case, minimization is handled using numerical iterative algorithms such as Newton-Raphson or Levenberg-Marquardt. The main advantage of these approaches are their accuracy. The main drawback is that they may be subject to local minima and, worse, divergence.

In this paper, pose computation is formulated in terms of a full scale non-linear optimization: Virtual Visual Servoing (VVS). In this way the pose computation problem is considered as similar to 2D visual servoing as proposed in [22], [21]. The new method presented in this paper is aligned with state of the art methods treating this issue [7], [18]. Essentially, 2D visual servoing [14], [8], [12] consists of specifying a task (mainly positioning or target tracking tasks) as the regulation in the image of a set of visual features. A closed-loop control law that minimizes the error between the current and desired position of these visual features can then be implemented which automatically determines the motion the camera has to realize. This framework is used to create an image feature based system which is capable of treating complex scenes in real-time. Advantages of the virtual visual servoing formulation are demonstrated by considering a wide range of performance factors. Notably the accuracy, efficiency, stability, and robustness issues have been addressed and demonstrated to perform in complex scenes. In particular, the interaction matrices that link the virtual camera velocity to the variation of a distance in the image are determined. A robust control law that integrates an M-estimator is integrated to improve robustness. The resulting pose computation algorithm is thus able to deal efficiently with incorrectly tracked features that usually contribute to a compound effect which degrades the system until failure.

In the remainder of this paper, Section II-A presents the principle of the approach. In Section II-B the details of the robust visual servoing control law are shown and a stability analysis is presented. In Section II-C the computation of the confidence in the local features extraction is introduced. Section III deals with the chosen visual features considered in the tracking process and the algorithm used for tracking local features is presented. In order to validate this approach the tracker is used as input to a 2D 1/2 visual servoing experiments. In section IV, several experimental results are reported.

II. Robust virtual visual servoing

A. Overview and motivations

As already stated, the fundamental principle of the proposed approach is to define the pose computation problem as the dual problem of 2D visual servoing [8], [14]. In visual servoing, the goal is to move a camera in order to observe an object at a given position in the image. An explanation will now be given as to why the pose computation problem is very similar.

To illustrate the principle, consider the case of an object with various 3D features ${}^o\mathbf{P}$ (for instance, ${}^o\mathbf{P}$ are the 3D coordinates of these features in the object frame). A virtual camera is defined whose position in the object frame is defined by $\mathbf{r}$. The approach consists of estimating the real pose by minimizing the error Δ between the observed data $\mathbf{s}^*$ (usually the position of a set of features in the image) and the position $\mathbf{s}$ of the same features computed by forward-projection according to the current pose:

$$\Delta = \big(\mathbf{s}(\mathbf{r}) - \mathbf{s}^*\big) = \big[pr_\xi(\mathbf{r},{}^o\mathbf{P}) - \mathbf{s}^*\big], \tag{1}$$

where $pr_\xi(\mathbf{r},{}^o\mathbf{P})$ is the projection model according to the intrinsic parameters ξ and camera pose $\mathbf{r}$. It is supposed here that intrinsic parameters ξ are available but it is possible, using the same approach, to also estimate these parameters.

In this formulation of the problem, a virtual camera initially at $\mathbf{r}_i$ is moved using a visual servoing control law in order to minimize the error Δ. At convergence, the virtual camera reaches the pose $\mathbf{r_d}$ which minimizes the error ($\mathbf{r_d}$ will be the real camera's pose).

An important assumption is to consider that $\mathbf{s}^*$ is computed (from the image) with sufficient precision. In visual servoing, the control law that performs the minimization of Δ is usually handled using a least squares approach [8][14]. However, when outliers are present in the measures, a robust estimation is required. M-estimators can be considered as a more general form of maximum likelihood estimators [13]. They are more general because they permit the use of different minimization functions not necessarily corresponding to normally distributed data. Many functions have been proposed in the literature which allow uncertain measures to be less likely considered and in some cases completely rejected. In other words, the objective function is modified to reduce the sensitivity to outliers. The robust optimization problem is then given by:

$$\Delta_{\mathcal{R}} = \rho\Big(\mathbf{s}(\mathbf{r}) - \mathbf{s}^*\Big), \tag{2}$$

where $\rho(u)$ is a robust function [13] that grows subquadratically and is monotonically nondecreasing with increasing $|u|$. Iteratively Re-weighted Least Squares (IRLS) is a common method of applying the M-estimator. It converts the M-estimation problem into an equivalent weighted least-squares problem.

To make use of robust minimization procedures in visual servoing a modification of the control law is required in order to reject outliers. Such a robust control law has been presented in a recent paper [4] within a different context. In [4] it is shown that a robust control law simultaneously accomplishes a visual servoing positioning task while remaining robust to a general class of image processing errors. In this paper the duality of the robust control law is used to perform pose estimation.

B. Robust control law

The objective of the control scheme is to minimize the objective function given in equation (2). This new objective is incorporated into the control law in the form of a weight which is given to specify a confidence in each feature location. Thus, the error to be regulated to 0 is defined as:

$$\mathbf{e} = \mathbf{D}(\mathbf{s}(\mathbf{r}) - \mathbf{s}^*), \tag{3}$$

where $\mathbf{D}$ is a diagonal weighting matrix given by $\mathbf{D} = diag(w_1, \ldots, w_k)$. The computation of weights w_i is described in Section II-C.

A simple control law can be designed to try and ensure an exponential decoupled decrease of $\mathbf{e}$ around the desired position $\mathbf{s}^*$ (see [4] for more details). The control law is given by:

$$\mathbf{v} = -\lambda(\widehat{\mathbf{D}\mathbf{L}_\mathbf{s}})^+\mathbf{D}\big(\mathbf{s}(\mathbf{r}) - \mathbf{s}^*\big), \tag{4}$$

where $\mathbf{v}$ is the virtual camera velocity.

A classical approach in visual servoing considers the Jacobian to be constant and it is calculated from the desired depth $\mathbf{Z}^*$ and the desired value of the features $\mathbf{s}^*$. In the VVS case, the desired depth is unknown but the initial value of $\mathbf{Z_i}$ is available, thus $(\widehat{\mathbf{D}\mathbf{L}_\mathbf{s}})^+$ can be defined as:

$$(\widehat{\mathbf{D}\mathbf{L}_\mathbf{s}})^+ = \mathbf{L}_\mathbf{s}^+(\mathbf{s_i}, \mathbf{Z_i}), \tag{5}$$

Clearly, it is also necessary to ensure that a sufficient number of features will not be rejected so that $\mathbf{D}\mathbf{L}_\mathbf{s}$ is always of full rank (6 to estimate the pose).

It has been shown that only local stability can be demonstrated [4]. This means that the convergence may not be obtained if the error $\mathbf{s} - \mathbf{s}^*$ is too large. However, in tracking applications $\mathbf{s}$ and $\mathbf{r}$ are obtained from the previous image, thus the motion between two successive images acquired at video rate is sufficiently small to ensure the convergence. In practice it has been observed that the convergence is obtained, in general, when the camera displacement has an orientation error less that 30^o on each axis. Thus, potential problems only appear for the very first image where the initial value for $\mathbf{r}$ may be too coarse. In the current algorithm the initialization is done by manually clicking on the images and calculating the pose using a 4 point algorithm [5].

C. Computing confidence

This section gives a brief overview for the calculation of weights for each image feature. The weights w_i, which represent the different elements of the $\mathbf{D}$ matrix and reflect the confidence of each feature [13], are usually given by:

$$w_i = \frac{\psi(\delta_i/\sigma)}{\delta_i/\sigma}, \tag{6}$$

where $\psi\Big(u\Big) = \frac{\partial\rho\Big(u\Big)}{\partial \mathbf{s}}$ (ψ is the influence function) and δ_i is the normalized residue given by $\delta_i = \Delta_i - Med(\Delta)$ (where $Med(\Delta)$ is the median operator) and σ is the standard deviation of the inlier data.

Of the various loss and corresponding influence functions that exist in the literature Tukey's hard re-descending function is considered. Tukey's function completely rejects outliers and gives them a zero weight. This is of interest in tracking applications so that a detected outlier has no effect on the virtual camera motion. This influence function is given by:

$$\psi(u) = \begin{cases} u(C^2 - u^2)^2 & , \text{ if } |u| \leq C \\ 0 & , \text{else,} \end{cases} \tag{7}$$

where the proportionality factor for Tukey's function is $C = 4.6851$ and represents 95% efficiency in the case of Gaussian Noise.

In order to obtain a robust objective function, a value describing the certainty of the measures is required. The scale σ that is the estimated standard deviation of the inlier data, is an important value for the efficiency of the method. In non-linear regression for pose computation, this estimate of the scale can vary dramatically during convergence. Scale may be manually chosen as a tuning variable or may be estimated online. In our case, we use a robust statistic to estimate scale with the Median Absolute Deviation (MAD), given by:

$$\widehat{\sigma} = \frac{1}{\Phi^{-1}(0.75)} Med_i(|\delta_i - Med_j(\delta_j)|). \tag{8}$$

where $\Phi()$ is the cumulative normal distribution function and $\frac{1}{\Phi^{-1}(0.75)} = 1.48$ represents one standard deviation of the normal distribution.

III. Visual features

A. Interaction matrices

Any kind of geometrical feature can be considered within the proposed control law as soon as it is possible to compute its corresponding interaction matrix $\mathbf{L_s}$. In [8], a general framework to compute $\mathbf{L_s}$ is proposed. Indeed, it is possible to compute the pose from a large set of image information (points, lines, circles, quadratics, distances, etc...) within the same framework. The combination of different features is achieved by adding features to vector $\mathbf{s}$ and by "stacking" each feature's corresponding interaction matrix into a large interaction matrix of size $nd \times 6$ where n corresponds to the number of features and d their dimension:

$$\begin{bmatrix} \mathbf{e}_1 \\ \vdots \\ \mathbf{e}_n \end{bmatrix} = \begin{bmatrix} \mathbf{L}_1 \\ \vdots \\ \mathbf{L}_n \end{bmatrix} \mathbf{v} \tag{9}$$

The redundancy yields a more accurate result with the computation of the pseudo-inverse of $\mathbf{L_s}$ as given in equation (4). Furthermore if the number or the nature of visual features is modified over time, the interaction matrix $\mathbf{L_s}$ and the vector error $\mathbf{s}$ is easily modified consequently.

In this paper, a distance feature is considered as a set of distances between local point features obtained from a fast image processing step and the contours of a more global CAD model. In this case the desired value of the distance is equal to zero. The assumption is made that the contours of the object in the image can be described as piecewise linear segments. All distances are then treated according to their corresponding segment.

The derivation of the interaction matrix that links the variation of the distance between a fixed point and a moving straight line to the virtual camera motion is now given. In Figure 1 $\mathbf{p}$ is the tracked point feature position and $\mathbf{l}(\mathbf{r})$ is the line feature position. The position of the

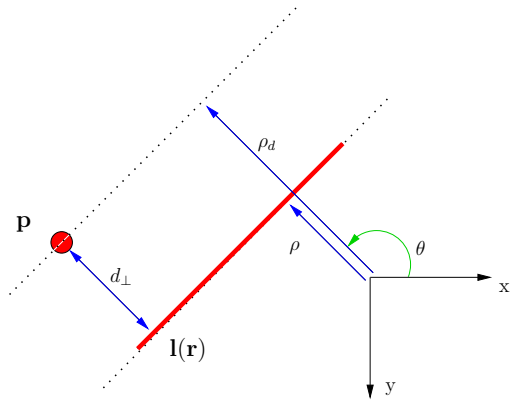

Fig. 1. Distance of a point to a line

line is given by its polar coordinates representation,

$$x\cos\theta + y\sin\theta = \rho, \forall (x,y) \in \mathbf{l}(\mathbf{r}), \tag{10}$$

The distance between a point $\mathbf{p}$ corresponding to a line $\mathbf{l}(\mathbf{r})$ is characterized fully by the distance $d_\perp$ perpendicular to the line. The point found to the normal of the line does not necessarily correspond to the departure point on the line for the edge detection procedure. In other words the distance parallel to the segment does not hold any useful information unless a correspondence exists between a point on the line and $\mathbf{p}$ (which is in general not the case). Thus the distance feature from a line is given by:

$$d_l = d_\perp(\mathbf{p}, \mathbf{l}(\mathbf{r})) = \rho(\mathbf{l}(\mathbf{r})) - \rho_d, \tag{11}$$

where

$$\rho_d = x_d\cos\theta + y_d\sin\theta, \tag{12}$$

with x_d and y_d being the coordinates of the tracked point. Thus, the variation of the distance with time can be related to the variation of the line parameters by:

$$\dot{d}_l = \dot{\rho} - \dot{\rho_d} = \dot{\rho} + \alpha\dot{\theta}, \tag{13}$$

where $\alpha = x_d\sin\theta - y_d\cos\theta$.

Using the relation (13), the interaction matrix for a distance feature d_l can be shown to be composed of the interaction matrix for a line:

$$\mathbf{L}_{d_l} = \mathbf{L}_\rho + \alpha\mathbf{L}_\theta. \tag{14}$$

The interaction matrix related to a straight line is given by (see [8] for its complete derivation):

$$\mathbf{L}_\theta = \begin{pmatrix} \lambda_\theta \cos\theta & \lambda_\theta \sin\theta & -\lambda_\theta \rho & \rho\cos\theta & -\rho\sin\theta & -1 \end{pmatrix}$$
$$\mathbf{L}_\rho = \begin{pmatrix} \lambda_\rho \cos\theta & \lambda_\rho \sin\theta & -\lambda_\rho \rho & (1+\rho^2)\sin\theta & -(1+\rho^2)\cos\theta & 0 \end{pmatrix}$$

where $\lambda_\theta = (A_2 \sin\theta - B_2 \cos\theta)/D_2$, $\lambda_\rho = (A_2 \rho \cos\theta + B_2 \rho \sin\theta + C_2)/D_2$, and $A_2X + B_2Y + C_2Z + D_2 = 0$ is the equation of a 3D plane which the line belongs to.

By evaluating (14) the following is obtained:

$$\mathbf{L}_{d_l} = \begin{bmatrix} \lambda_{d_l} \cos\theta \\ \lambda_{d_l} \sin\theta \\ -\lambda_{d_l} \rho \\ (1+\rho^2)\sin\theta - \alpha\rho\cos\theta \\ -(1+\rho^2)\cos\theta - \alpha\rho\sin\theta \\ -\alpha \end{bmatrix}^T, \qquad (15)$$

where $\lambda_{d_l} = \lambda_\rho + \alpha\lambda_\theta$.

Note that the case of a distance between a point and the projection of a cylinder or a portion of an ellipse is similar [3].

B. Tracking visual features

When dealing with low-level image processing, the object model is projected onto the image and the contours are sampled at a regular distance. At these sample points a 1 dimensional search is performed to the normal of the contour for corresponding edges. An *oriented* gradient mask [1] is used to detect the presence of a similar contour. One of the advantages of this method is that it only searches for edges which are aligned in the same direction as the parent contour. An array of 180 masks is generated off-line which is indexed according to the contour angle. This is therefore implemented with convolution efficiency, and leads to real-time performance.

More precisely, the process consists of searching for the corresponding point p^{t+1} in image I^{t+1} for each point p^t (see Figure 2). A 1D search interval $\{Q_j, j \in [-J, J]\}$ is determined in the direction δ of the normal to the contour. For each position Q_j lying in the direction δ a mask convolution M_δ corresponding to the square root of a log-likelihood ratio ζ_j is computed. Then the new position p^{t+1} is given by:

$$Q^{j^*} = \arg \max_{j \in [-J,J]} \zeta_j \text{ with } \zeta_j = \mid I^{t+1}_{\nu(Q_j)} * M_\delta \mid$$

$\nu(.)$ is the neighborhood of the considered pixel. In this paper the neighborhood is limited to a 7×7 pixel mask. It should be noted that there is a trade-off to be made between real-time performance and mask stability. Likewise there is a trade-off to be made between the search distance, real-time performance while considering the maximum inter-frame movement of the object.

This low level search produces a list of k points which are used to calculate distances from corresponding projected contours.

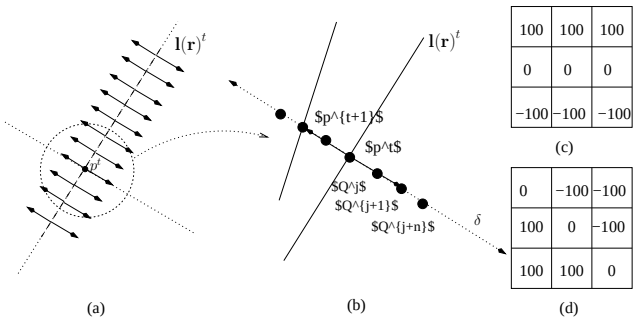

Fig. 2. Determining points position in the next image using the oriented gradient algorithm: (a) calculating the normal at sample points, (b) sampling along the normal (c-d) 2 out of 180 3x3 predetermined masks (in practice 7x7 masks are used) (c) 180^o (d) 45^o.

IV. Experimental results

Any current visual servoing control law can be implemented using the presented tracker (image-based, position-based or hybrid scheme) because all 3D pose information has been estimated. In the following experiments the 2D 1/2 visual servoing approach is used [2].

The visual feature vector $\mathbf{s}$ is selected as $(\mathbf{T}, x, y, \theta U_z)$ where $\mathbf{T}$, expressed in the desired camera frame, is the translation that the camera has to realize, x and y are the coordinates of an image point, and θU_z is the third component of vector $\theta\mathbf{U}$ (where θ and $\mathbf{U}$ are the angle and the axis of the rotation that the camera has to realize). Using this control law, the camera translation is specified such that it is a straight line in the Cartesian space (which is a particularly satisfactory trajectory), and camera pan and tilt are constrained such that the trajectory of the selected point is a straight line in the image. The interaction matrix related to $\mathbf{s}$ is given by [2]:

$$\mathbf{L_s} = \begin{pmatrix} \mathbf{R} & \mathbf{0_3} \\ \frac{1}{Z}\mathbf{L}_{v\omega} & \mathbf{L}_\omega \end{pmatrix} \qquad (16)$$

where $\mathbf{R}$ is the rotation matrix from current to desired camera frames and:

$$\mathbf{L}_{v\omega} = \begin{pmatrix} -1 & 0 & x \\ 0 & -1 & y \\ 0 & 0 & 0 \end{pmatrix} \text{ and } \mathbf{L}_\omega = \begin{bmatrix} xy & -(1+x^2) & y \\ (1+y^2) & -xy & -x \\ l_1 & l_2 & l_3 \end{bmatrix}$$

(l_1, l_2, l_3) being the third row of matrix given by:

$$\mathbf{I_3} - \frac{\theta}{2}\tilde{\mathbf{U}} + \left(1 - \frac{\text{sinc}(\theta)}{\text{sinc}^2(\frac{\theta}{2})}\right)\tilde{\mathbf{U}}^2 \qquad (17)$$

with $\text{sinc}(\theta) = \sin(\theta)/\theta$, $\tilde{\mathbf{U}}$ being the antisymmetric matrix associated to $\mathbf{U}$.

This interaction matrix is never singular except in degenerate cases and the following control scheme is applied:

$$\mathbf{v} = -\lambda {\mathbf{L_s}}^{-1}(\mathbf{s} - \mathbf{s_d}) \qquad (18)$$

where λ is an adaptive gain that is function of the error $\mathbf{s} - \mathbf{s}^*$ and is tuned in order to speed-up convergence.

The complete implementation of the robust visual servoing task, including tracking and control, was carried out on an experimental test-bed involving a CCD camera

mounted on the end effector of a six d.o.f robot. Images were acquired and processed at video rate (50Hz). In such experiments, the image processing is potentially very complex. Indeed, extracting and tracking reliable points in real environment is a non trivial issue. In all experiments, the distances are computed using the "oriented" gradient mask algorithm described previously. Tracking is always performed at below frame rate (usually in less than 10ms).

All the figures depict the current position of the tracked object in green while its desired position appears in blue. Three objects where considered: a micro-controller (Figure 3), an industrial emergency switch (Figure 4) and a video multiplexer (Figure 5).

To validate the robustness of the algorithm, the micro-controller was placed in a highly textured environment as shown in Fig. 3. Tracking and positioning tasks were correctly achieved. Multiple temporary and partial occlusions were made by a hand and various work-tools as well as modification of the lighting conditions were imposed during the realization of the positioning task. In the second experiment the robot velocity reaches a velocity of 23 cm/s in translation and 85 deg/s in rotation. Thus, less than 35 frames were acquired during the entire positioning task up until convergence (see Figure 4e). On the third experiment (Figure 5) after a complex positioning task (note that some object faces appeared while others disappeared) the object is handled by hand and moved around. In this case, since the visual servoing task has not been stopped, the robot continues to follow the object in order to maintain the rigid link between the camera and the object.

For the latter two experiments, plots are also shown which give an analysis of the pose parameters, the camera velocity and the error vector $(\mathbf{s} - \mathbf{s}^*)$. In other words, the task was accomplished in less than 1 second. In all these experiments, neither a Kalman filter (or other prediction process) or the camera displacement were used to help the tracking.

V. Conclusion and Future Perspectives

A method has been presented for robustly tracking complex objects in an image sequence at a high processing rate (50Hz) using virtual visual servoing techniques. High robustness has been demonstrated with a robust model-based approach using an iteratively re-weighted least squares implementation of a M-estimator. The use of a non-linear minimization approach coupled with a redundant number of distance-to-contour visual features leads to efficiency and robustness. The presented method allows fast and accurate positioning of a eye-in-hand robot with respect to real objects (without any landmarks) in complex situations. The algorithm has been tested on various real visual servoing scenarios demonstrating a real usability of this approach.

References

[1] P. Bouthemy. A maximum likelihood framework for determining moving edges. *IEEE Trans. on Pattern Analysis and Machine intelligence*, 11(5):499–511, May 1989.

[2] F. Chaumette and E. Malis. 2 1/2 D visual servoing: a possible solution to improve image-based and position-based visual servoings. In *IEEE Int. Conf. on Robotics and Automation*, vol. 1, pp. 630–635, San Francisco, April 2000.

[3] A. Comport, E. Marchand, and F. Chaumette. A real-time tracker for markerless augmented reality. In *ACM/IEEE Int. Symp. on Mixed and Augmented Reality, ISMAR'03*, pp. 36–45, Tokyo, Japan, October 2003.

[4] A. Comport, M. Pressigout, E. Marchand, and F. Chaumette. A visual servoing control law that is robust to image outliers. In *IEEE Int. Conf. on Intelligent Robots and Systems, IROS'03*, vol. 1, pp. 492–497, Las Vegas, October 2003.

[5] D. Dementhon and L. Davis. Model-based object pose in 25 lines of codes. *Int. J. of Computer Vision*, 15:123–141, 1995.

[6] M. Dhome, M. Richetin, J.-T. Lapresté, and G. Rives. Determination of the attitude of 3D objects from a single perspective view. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 11(12):1265–1278, December 1989.

[7] T. Drummond and R. Cipolla. Real-time visual tracking of complex structures. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 27(7):932–946, July 2002.

[8] B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. *IEEE Trans. on Robotics and Automation*, 8(3):313–326, June 1992.

[9] D.B. Gennery. Visual tracking of known three-dimensional objects. *Int. J. of Computer Vision*, 7(3):243–270, 1992.

[10] G. Hager and K. Toyama. The XVision system: A general-purpose substrate for portable real-time vision applications. *Computer Vision and Image Understanding*, 69(1):23–37, January 1998.

[11] R. Haralick, H. Joo, C. Lee, X. Zhuang, V Vaidya, and M. Kim. Pose estimation from corresponding point data. *IEEE Trans on Systems, Man and Cybernetics*, 19(6):1426–1445, November 1989.

[12] K. Hashimoto, editor. *Visual Servoing : Real Time Control of Robot Manipulators Based on Visual Sensory Feedback*. World Scientific Series in Robotics and Automated Systems, Vol 7, World Scientific Press, Singapor, 1993.

[13] P.-J. Huber. *Robust Statistics*. Wiler, New York, 1981.

[14] S. Hutchinson, G. Hager, and P. Corke. A tutorial on visual servo control. *IEEE Trans. on Robotics and Automation*, 12(5):651–670, October 1996.

[15] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. In *European Conference on Computer Vision, ECCV'96, LNCS no. 1064, Springer-Verlag*, pp. 343–356, Cambridge, 1996.

[16] C. Kervrann and F. Heitz. A hierarchical Markov modeling approach for the segmentation and tracking of deformable shapes. *Graphical Models and Image Processing*, 60(3):173–195, May 1998.

[17] H. Kollnig and H.-H. Nagel. 3D pose estimation by fitting image gradients directly to polyhedral models. In *IEEE Int. Conf. on Computer Vision*, pp. 569–574, Boston, May 1995.

[18] D. Kragic and H.I. Christensen. Confluence of parameters in model based tracking. In *IEEE Int. Conf. on Robotics and Automation, ICRA'03*, vol. 4, pp. 3485–3490, Taipe, Taiwan, September 2003.

[19] D.G. Lowe. Fitting parameterized three-dimensional models to images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(5):441–450, May 1991.

[20] C.P. Lu, G.D. Hager, and E. Mjolsness. Fast and globally convergent pose estimation from video images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(6):610–622, June 2000.

[21] E. Marchand and F. Chaumette. Virtual visual servoing: a framework for real-time augmented reality. In *EUROGRAPHICS'02 Conference Proceeding*, vol. 21(3) of *Computer Graphics Forum*, pp. 289–298, Saarebrücken, Germany, September 2002.

[22] V. Sundareswaran and R. Behringer. Visual servoing-based augmented reality. In *IEEE Int. Workshop on Augmented Reality*, San Francisco, November 1998.

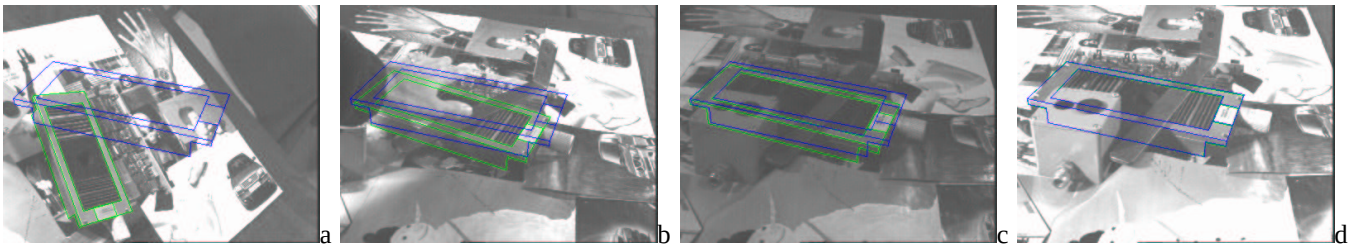

Fig. 3. Tracking in a complex environment for a classical visual servoing experiment: Images are acquired and processed at video rate (25Hz). Blue: desired position defined by the user. Green: position measured after pose calculation. (a) first image initialized by hand, (b) partial occlusion with hand, (c) lighting variation, (d) final image with various occlusions

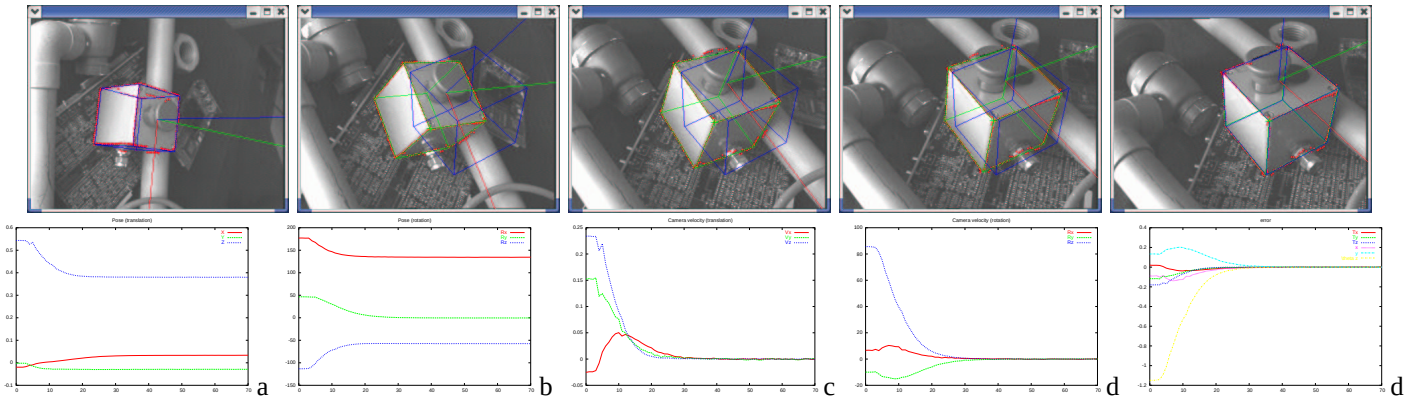

Fig. 4. 2D 1/2 visual servoing experiments: on these five snapshots the tracked object appears in green and its desired position in the image in blue. Plots correspond to (a) Pose (translation) (b) Pose (rotation) (c-d) camera velocity in rotation and translation (e) error vector $\mathbf{s} - \mathbf{s}^*$

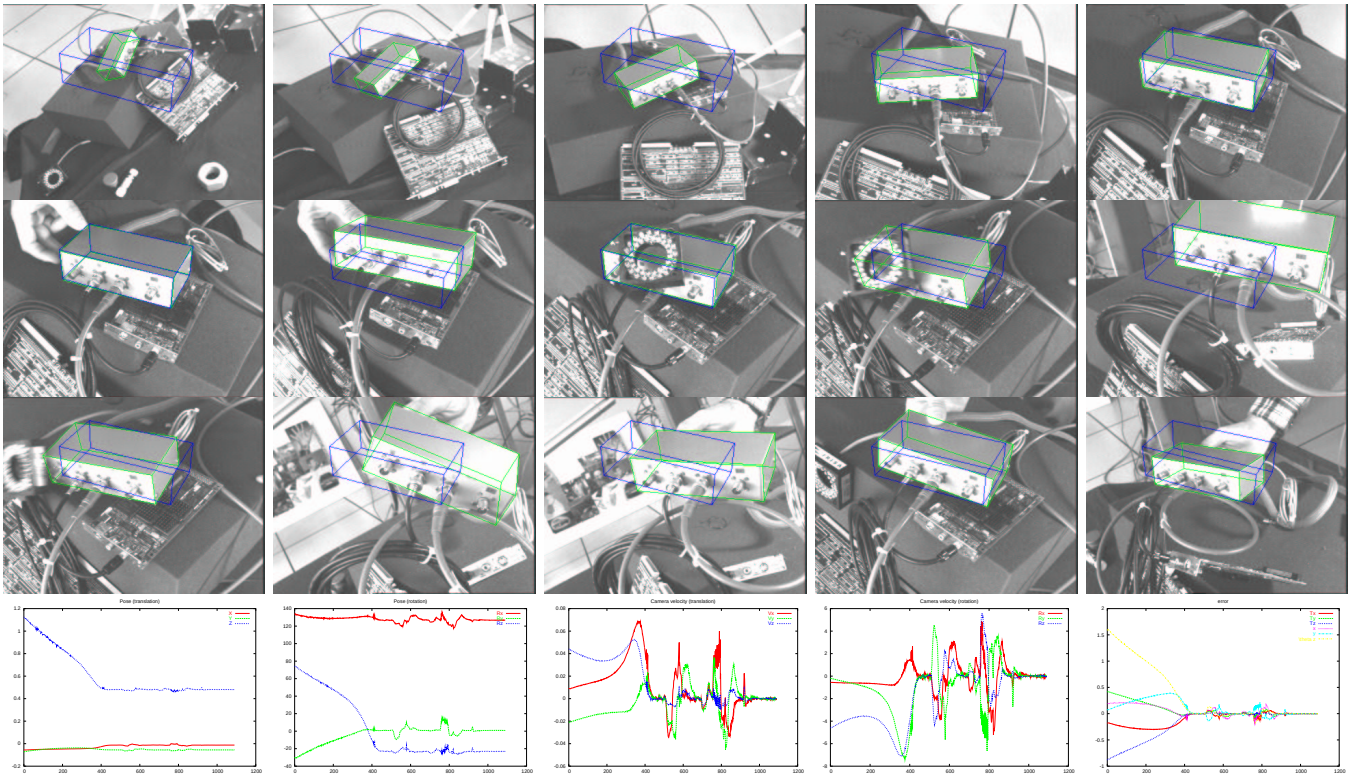

Fig. 5. 2D 1/2 visual servoing experiments: in these images the tracked object appears in green and its desired position in the image in blue. The six first images have been acquired in initial visual servoing step. In the reminder images object is moving along with the robot. Plots corresponds to (a) Pose (translation) (b) Pose (rotation) (c-d) camera velocity in rotation and translation (e) error vector $\mathbf{s} - \mathbf{s}^*$